## Remarks

Reconsideration of the application is respectfully requested in view of the foregoing amendments and following remarks. Claims 1-23 are pending in the application. No claims have been allowed. Claims 1, 8, 13, and 17 are independent. Claims 1 and 8 have been amended and claim 9 has been canceled.

### Provisional Double Patenting Rejection

Claims 1-23 stand provisionally rejected over claims 1-12 of co-pending Application No. 09/432,854 under the judicially created doctrine of double patenting. Applicants submit herewith a Terminal Disclaimer. Accordingly, Applicants respectfully request the double patenting rejection be withdrawn.

### Cited Art

The Action cites U.S. Pat. No. 6,389,464 to Krishnamurthy et al. ("*Krishnamurthy*") and U.S. Pat. No. 6,658,464 to Reisman ("*Reisman*").

### Section 103 Rejections

To establish a prima facie case of obviousness, three basic criteria must be met. First, there must be some suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify the reference or to combine reference teachings. Second, there must be a reasonable expectation of success. Finally, the prior art reference (or references when combined) must teach or suggest all the claim limitations. (MPEP § 2142.)

Motivations to combine or modify references must come from the references themselves or be within the body of knowledge in the art. (*See,* MPEP § 2143.01.)

### Patentability of claims 1-23 over *Krishnamurthy* in view of *Reisman* under § 103(a)

The Action rejects claims 1-23 under 35 U.S.C § 103(a) as unpatentable over *Krishnamurthy* in view of *Reisman*. The Applicants respectfully traverse the rejection. The claims in their present form should be allowed over the cited art.

Independent Claim 1

The now amended Claim 1 recites as follows:

> A method of programmatically controlling a service of a logical device realized on a first computer from a second computer, the method comprising:
>
> from the first computer, obtaining at the second computer a service description message related to the service, the service description message detailing a set of actions that can be invoked on the service via network data messages conveyed to the first computer via peer-to-peer networking connectivity over a data communications network connecting the first and the second computer;
>
> exposing a programming interface to access by software programs running on the second computer, the programming interface having an action-invoking member;
>
> based on the service description message, converting a programmatic invocation of the action-invoking member of the programming interface by a software program running on the second computer into a network data message for invoking an action of the service via peer-to-peer networking connectivity over the data communications network; and
>
> transmitting the network data message to the first computer to thereby invoke the action of the service.

*Krishnamurthy fails to teach or suggest "from the first computer, obtaining at the second computer a service description message related to the service, the service description message detailing a set of actions that can be invoked on the service via network data messages conveyed to the first computer via peer-to-peer networking connectivity over a data communications network connecting the first and the second computer."* The Action relies on the web manager and the site server of the network management system taught by *Krishnamurthy*. However, according to what is taught by *Krishnamurthy*, neither the web manager nor the site server is the "second computer" which is operable for "from the first computer, obtaining at the second computer a service description message related to the service, the service description message detailing a set of actions that can be invoked on the service" further wherein the "service" is a "a service of a logical device realized on a first computer." As recited by claim 1, "service description message" is sent "from the first computer" which also hosts "a logical device" whose "service" is described in the "service description message" being sent. This is not taught or suggested by *Krishnamurthy*.

The Action does not specifically allege as to which elements of *Krishnamurthy's* network (e.g., the site server, the web manager and the remote computer) discloses the "first computer" or "the second computer" for performing the method as claimed in claim 1. Nevertheless, none of these elements of *Krishnamurthy's* network are operable for sending a "service description message" regarding "a service" that may available therein to any "second computer" as recited in claim 1.

Thus, conversely, it is also true that neither the site server nor the web manager nor the remote computer taught by *Krishnamurthy* is the "second computer" which is operable for "obtaining at the second computer a service description message related to the service."

*Reisman's description of a generic client user interface for use with any one or more of many online server-based information distribution services also fails to teach or suggest "from the first computer, obtaining at the second computer a service description message related to the service, the service description message detailing a set of actions that can be invoked on the service." Reisman* teaches a method of providing a generic user interface for client computers in a client-server network to communicate with different servers associated with different "information distribution services." According to *Reisman*, client computers in a client-server network may comprise an allegedly flexible client interface that uses an API to "communicate" a user's functional requests to a variety of servers associated with various internet service providers with different proprietary protocols. However, nothing in *Reisman* teaches or suggests that the client computers or the server computers of *Reisman* are the "second computer" which is operable for "from the first computer, obtaining at the second computer a service description message related to the service, the service description message detailing a set of actions that can be invoked on the service." (Emphasis added.) At best, the client computers may provide access to data available on the server computers. However, that does not lead one to a "second computer" which is operable for "obtaining... a service description message related to the service, the service description message detailing a set of actions that can be invoked on the service" wherein the "service description message" is "from the first computer" which also comprises the "service" that is the subject of the "service description message." This is true at least, because neither client computer nor the server recited in *Reisman* send or receive a "service description message" as recited in claim 1.

*Furthermore, the generic client interface API of Reisman used for communicating user requests to any one of a set of online services with different proprietary protocols does not teach or suggest "based on the service description message, converting a programmatic invocation of the action-invoking member of the programming interface by a software program running on the second computer into a network data message for invoking an action of the service via peer-to-peer networking connectivity over the data communications network."* (Emphasis added.) The client interface API taught by *Reisman* simply allows access to "translators and protocol drivers capable of communicating the user's functional requests to any one of a set of online services, using their

corresponding proprietary protocols." See *Reisman*, Col. 23, Lns. 58-61. The use of such an API is not "based on the service description message" as recited in claim 1. In fact, the API of *Reisman* available on a client computer is not based on any message sent by another computer. More particularly, the generic API of *Reisman* does not lead one to "based on the service description message, converting a programmatic invocation of the action-invoking member of the programming interface by a software program running on the second computer into a network data message for invoking an action of the service" particularly wherein the "service description message" is obtained "from the first computer" and is related to a "service" which is "of a logical device realized on a first computer" as recited in claim 1.

At least for the reasons listed above, the cited references, individually or when combined, fail to describe at least one feature recited in claim 1. Thus, claim 1 in its present form should be allowed over the cited art.

Claims 2-4

Claims 2-4 either directly or indirectly depend on claim 1. Thus, at least for the reasons listed above with respect to claim 1, claims 2-4 in their present form should also be allowed over the cited art.

Claim 5

Claim 5 depends directly on claim 1 should be allowed over the combination of *Krishnamurthy* and *Reisman* for at least the reasons listed above with respect to claim 1 and because of the independently patentable combination of method acts set forth therein. For example, claim 5 recites:

> The method of claim 1 wherein the action-invoking member accepts an invocation parameter indicating the action of the service that is to be invoked.

Thus, claim 5 refers to an "action-invoking member" of a "the programming interface having an action-invoking member" which "accepts an invocation parameter indicating the action of the service that is to be invoked." The Action relies on *Krishnamurthy* which teaches forms "used to allow the user to enter or modify configuration data that is stored in database" thereby allowing "a single ...Web manager 18 to manage different kinds of devices." See *Krishnamurthy* at Col. 8, Lns. 49-51. First of all, the Action is not clear as to what in *Krishnamurthy* teaches or suggests an

"action-invoking member" particularly one comprised within a "programming interface" as recited in claims 1 and 5. This is especially confusing in light of the Action's own admission that *Krishnamurthy* does not teach or suggest a "programming interface." See e.g., Action at page 4 stating "Krishnamurthy does not explicitly disclose exposing a programming interface to access by software programs running on the second computer, the programming interface having an action-invoking member."

Even if for argument's sake, *Krishnamurthy* did teach an "action-invoking member" it does not teach or suggest "wherein the action-invoking member accepts an invocation parameter indicating the action of the service that is to be invoked." The Action relies on user entered forms to "modify configuration data" for enabling communication with a particular managed network device. However, "configuration data" as taught by *Krishnamurthy* comprises "specifying a managed device driver, and selecting from port configuration options, and port drivers, alert messaging options, and MIB file administration options" none of which teach or suggest "an invocation parameter indicating the action of the service that is to be invoked." See e.g., *Krishnamurthy* at Col. 4, Lns. 57-60 for description of "configuration data." On the other hand the specification at pages 29-30 describes at least one embodiment of an action invocation method (e.g., InvokeServiceAction()) that can be called on a service object available on a "first computer" wherein an exemplary "invocation parameter indicating the action of the service that is to be invoked" is described as "SERVICE_ACTION." No such invocation parameters are taught or suggested by *Krishnamurthy*, *Reisman* or a combination thereof.

Claim 6

Claim 6 recites:

> The method of claim 1 wherein the programming interface further has a service state-querying member, the method further comprising:
> responsive to programmatic invocation of the service state-querying member by the software programs running on the second computer, obtaining state data of the service via peer-to-peer networking connectivity over the data communications network; and
> returning the state data to the invoking software program.

The Action rejects claim 6 based on the Examiner's Official Notice that it was "extremely well known in the networking art at the time [of] the invention to obtain state data regarding services on one system." The Applicants respectfully disagree and traverse the rejection based on Official

Notice. Claim 6 recites a method using a "programming interface" which "has a service state-querying member" which can be used by any program coded therein to invoke "the service state-querying member". This is much more flexible and independent than using "SNMP agents to obtain state information on other systems within the network." Thus, Applicants respectfully request the Examiner to provide documentary evidence showing that the claimed method acts were indeed well-known in the art. See MPEP § 2144.03 - C.

Claim 7

Claim 7 recites:

> The method of claim 1 wherein the programming interface further has a service state-querying member that accepts an invocation parameter indicative of a state data variable of the service, the method further comprising:
>
> responsive to programmatic invocation of the service state-querying member by the software programs running on the second computer, obtaining a value of the state data variable of the service via peer-to-peer networking connectivity over the data communications network; and
>
> returning a datum indicative of the value of the state data variable to the invoking software program.

The Action rejects claim 7 based on Examiner's Official Notice that it was "extremely well known in the networking art at the time [of] the invention to obtain state data regarding services on one system." The Applicants respectfully disagree and traverse the rejection. At least for the reasons listed above with respect to claim 6, Applicants respectfully request the Examiner to provide documentary evidence showing that the claimed method acts were indeed well-known in the art. See MPEP § 2144.03 - C.

Independent Claim 8

The now amended Claim 8 recites:

> In a networking environment providing peer-to-peer connectivity between logical devices on separate computing machines on a data communications network in accordance with a control protocol, the control protocol defining an exchange between a control point and a controlled logical device service in which the controlled logical device service furnishes a service description document to the control point, the service description document specifying a set of actions invocable on the controlled logical device service via peer networking data messages, the control point transmitting the peer networking data messages to the controlled logical device service to cause respective actions to be performed, a user-operated control device comprising:

a rehydrating module;

an application programming interface exposed by the rehydrating module to access from application software running on the user-operated control device, the application programming interface having an invoke action member; and

invoke action member-implementing code of the rehydrating module operating responsive to an invocation of the invoke action member to generate a peer networking data message to cause the controlled logical device service to perform a respective action of the controlled logical device service;

service description-obtaining code of the rehydrating module operating to obtain the service description document from the controlled logical device service per the control protocol;

and converting code of the rehydrating module operating to construct the peer networking data message based on the obtained service description document.

*Neither Krishnamurthy nor Reisman either individually or in combination teach or suggest "service description-obtaining code of the rehydrating module operating to obtain the service description document from the controlled logical device service per the control protocol; and converting code of the rehydrating module operating to construct the peer networking data message based on the obtained service description document."* The Action relies on a combination of *Krishnamurthy* and *Reisman* by stating that "claims 8-12 list all the same elements of claim 1-7, but in device form rather than method form. Therefore the supporting rationale of the rejection to claims 1-7 applies equally as well to claims 8-12." See Action, Page 5. Applicants respectfully disagree and note that claims 8-12 comprise elements not recited by claims 1-7.

Nevertheless, none of the elements of *Krishnamurthy's* network (e.g., the site server, the web manager and the remote computer) are operable to "to obtain the service description document from the controlled logical device service per the control protocol" wherein the "service description document" is regarding "a controlled logical device service" that may available on "a controlled logical device" as recited in claim 8. Thus, conversely, it is also true that neither the site server nor the web manager nor the remote computer taught by *Krishnamurthy* is the "the controlled logical device service" which is operable to send such a "service description document."

Furthermore, also as noted above, according to the Action, itself, *Krishnamurthy* fails to teach "an application programming interface exposed by the rehydrating module" as recited in claim 8. Also, the client interface API taught by *Reisman* simply allows access to "translators and protocol drivers capable of communicating the user's functional requests to any one of a set of online services, using their corresponding proprietary protocols." See *Reisman*, Col. 23, Lns. 58-61.

However, claim 8 recites an "application programming interface having an invoke action member" and a "rehydrating module operating responsive to an invocation of the invoke action member to generate a peer networking data message" further wherein "converting code of the rehydrating module operating to construct the peer networking data message based on the obtained service description document" wherein "the service description document" is one "specifying a set of actions invocable on the controlled logical device service." (Emphasis added.). As noted above, use of the generic API taught by *Reisman* is not "based on the obtained service description document" and in fact, as far as the Applicants can tell, *Reisman's* API is not based on any message sent by another computer.

At least for the reasons listed above, the cited references, individually or when combined, fail to describe at least one feature recited in claim 8. Thus, claim 8 in its present form should be allowed over the cited art.

Claims 10-12

Claims 10-12, either directly or indirectly, depend on claim 8. Thus, at least for the reasons set forth above with regard to claim 8, claims 10-12 should also be allowed.

Independent Claim 13

Claim 13 recites:

> A computer-readable data carrying medium having software program code carried thereon, the software program code comprising:
>
> a programmatic peer networking device service control module providing programmatic control by application software on a computing device executing the software program code of logical device services of separate computing devices on a data communications network via a peer-to-peer networking connectivity service control protocol;
>
> an application programming interface exposed by the programmatic peer networking device service control module for access by the application software, the application programming interface being a run-time dispatch interface having an invoke service action method member, the invoke service action method member accepting an action identifier, ingoing action arguments, outgoing action arguments, and action return value as parameters upon invocation by the application software; and
>
> invoke service action method member-implementing code of the programmatic peer networking device service control module operating responsive to an invocation of the invoke service action method member on the application programming interface by the application software to exchange data

messages with a logical device service of a separate computing device on the data communications network in accordance with the peer-to-peer networking connectivity service control protocol so as to invoke an action of the logical device service as per the parameters of the invoke service action method member and pass outgoing action arguments and action return value from the logical device service back to the application software.

*Neither Krishnamurthy nor Reisman either individually or in combination teach or suggest "an application programming interface exposed by the programmatic peer networking device service control module for access by the application software, ...the invoke service action method member accepting an action identifier, ingoing action arguments, outgoing action arguments, and action return value as parameters upon invocation by the application software."* The Action relies on a combination of *Krishnamurthy* and *Reisman* by stating that "claims 13-16 list all the same elements of claim 1-7, but in computer-readable data-carrying medium form rather than method form. Therefore, the supporting rationale of the rejection to claims 1-7 applies equally as well to claims 13-16." See Action, Page 5. The applicants respectfully disagree and note that claims 13-16 comprise elements not recited by claims 1-7.

Nevertheless, as noted with respect to claim 5 above, *Krishnamurthy* teaches forms to "modify configuration data" for enabling communication with a particular type of managed network device. First of all, according to the Action's itself, *Krishnamurthy* fails to teach or suggest "an application programming interface" recited by claim 13. See e.g., Action at page 4 stating "Krishnamurthy does not explicitly disclose exposing a programming interface to access by software programs running on the second computer, the programming interface having an action-invoking member." *Reisman* on the other hand, does teach an API which allows access to "translators and protocol drivers capable of communicating the user's functional requests to any one of a set of online services, using their corresponding proprietary protocols." See *Reisman*, Col. 23, Lns. 58-61. However, *Reisman's* API related to accessing protocol drivers and translators does not teach or suggest "an application programming interface exposed by the programmatic peer networking device service control module" comprising "the invoke service action method member accepting an action identifier, ingoing action arguments, outgoing action arguments, and action return value as parameters upon invocation by the application software" as recited in claim 13. (Emphasis added.) Combining *Krishnamurthy* with *Resiman* still does not lead one to "the invoke service action method member accepting an action identifier, ingoing action arguments, outgoing action arguments, and action return value as parameters upon invocation by the application software." (Emphasis

added.). This is so, at least because, the "configuration data" as taught by *Krishnamurthy* comprises "specifying a managed device driver, and selecting from port configuration options, and port drivers, alert messaging options, and MIB file administration options" none of which teach or suggest "the invoke service action method member accepting an action identifier, ingoing action arguments, outgoing action arguments, and action return value as parameters upon invocation by the application software" as recited by claim 13. (Emphasis added.)

At least for the reasons listed above, the cited references, individually or when combined, fail to describe at least one feature recited in claim 13. Thus, claim 13 in its present form should be allowed over the cited art.

Claims 14-16

Claims 14-16 depend on claim 13 either directly or indirectly and thus, at least for the reasons listed above with reference to claim 13, claims 14-16 should also be allowed.

Independent Claim 17

Claim 17 recites:

> A software module carried on a computer-executable software carrying medium, the software module exposing a programming interface for providing programmatic logical device service control via peer networking connectivity, the programming interface comprising:
> an invoke action method member having parameters for passing an action identifier, action arguments and action return value;
> wherein an implementation of the invoke action method member in the software module converts an invocation of the invoke action method member into an exchange of text messages with a logical device via peer networking connectivity based on a service description obtained from the logical device to control a service of the logical device.

The Action relies on a combination of *Krishnamurthy* and *Reisman* by stating that "claims 17-23 list all the same elements of claim 1-7, but in software module form rather than method form. Therefore, the supporting rationale of the rejection to claims 1-7 applies equally as well to claims 17-23." See Action, Page 6. The applicants respectfully disagree and note that claims 17-23 comprise elements not recited by claims 1-7.

Nevertheless, at least for the reasons listed above with respect to claim 13, neither *Krishnamurthy* nor *Reisman* either individually or in combination teach or suggest "programming

interface comprising: an invoke action method member having parameters for passing an action identifier, action arguments and action return value." (Emphasis added.)

Furthermore, neither *Krishnamurthy* nor *Reisman* either individually or in combination teach or suggest "wherein an implementation of the invoke action method member in the software module converts an invocation of the invoke action method member into an exchange of text messages with a logical device via peer networking connectivity based on a service description obtained from the logical device to control a service of the logical device." At least for the reasons listed above with respect to claims 1 and 8, none of the elements of *Krishnamurthy's* network (e.g., the site server, the web manager and the remote computer) teach or suggest "a service description obtained from the logical device to control a service of the logical device." *Reisman* does recite an API which allows access to "translators and protocol drivers capable of communicating the user's functional requests to any one of a set of online services, using their corresponding proprietary protocols." See *Reisman*, Col. 23, Lns. 58-61. However, at least for the reasons listed above with respect to claims 1 and 8 the combination of *Krishnamurthy* and *Reisman* still fails to teach or suggest "the software module converts an invocation of the invoke action method member into an exchange of text messages with a logical device via peer networking connectivity based on a service description obtained from the logical device to control a service of the logical device."

At least for the reasons listed above, the cited references, individually or when combined, fail to describe at least one feature recited in claim 17. Thus, claim 17 in its present form should be allowed over the cited art.

Claims 18-23

Claims 18-23 depend either directly or indirectly on claim 17 and thus, at least for the reasons listed above with respect to claim 17, claims 18-23 should also be allowed.

## Conclusion

The claims in their present form should now be allowable. Such action is respectfully requested.

Respectfully submitted,

KLARQUIST SPARKMAN, LLP

By _____
Stephen A. Wight
Registration No. 37,759

One World Trade Center, Suite 1600
121 S.W. Salmon Street
Portland, Oregon 97204
Telephone: (503) 226-7391
Facsimile: (503) 228-9446

(127693.3)